

# SKRIPSI

## IMPLEMENTASI ALGORITMA COUNTOUR ANALYS UNTUK PENGENALAN OBJEK ALFANUMERIK MENGGUNAKAN C#



*Disusun Oleh :*

**RONY HAMKA**  
**NIM: 05.12.669**

**KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA**  
**JURUSAN TEKNIK ELEKTRO S-1**  
**FAKULTAS TEKNOLOGI INDUSTRI**  
**INSTITUT TEKNOLOGI NASIONAL MALANG**  
**2012**

**LEMBAR PERSETUJUAN**

**IMPLEMENTASI ALGORITMA COUNTOUR ANALYS  
UNTUK PENGENALAN OBJEK ALFANUMERIK  
MENGUNAKAN C#**

*SKRIPSI*

*Disusun dan Diajukan Untuk Melengkapi dan Memenuhi Syarat  
Guna Mencapai Gelar Sarjana Teknik*

Disusun oleh :

**RONY HAMKA**


**NIM : 05.12.669**

**Diperiksa dan Disetujui,**

**Dosen Pembimbing I**

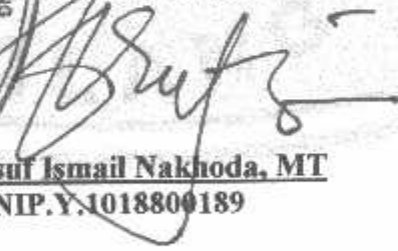
  
**Irmalia Suryani F. ST, MT**  
**NIP. P. 1030000365**

**Dosen Pembimbing II**

  
**Sandy Nataly Manja, S.KOM**  
**NIP.P. 1030800418**



**Mengetahui,  
Ketua Jurusan T. Elektro S-1**

  
**Ir. Yusuf Ismail Nakhoda, MT**  
**NIP.Y.1018800189**

**KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA  
JURUSAN TEKNIK ELEKTRO S-1  
FAKULTAS TEKNOLOGI INDUSTRI  
INSTITUT TEKNOLOGI NASIONAL MALANG  
2012**

## SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini :

Nama : Rony Hamka

NIM : 051266

Program Studi : Teknik Elektro SI


Konsentrasi : Teknik Informatika dan Komputer SI

Dengan ini menyatakan bahwa Skripsi yang saya buat adalah hasil karya sendiri, tidak merupakan plagiasi dari karya orang lain. Dalam Skripsi ini tidak memuat karya orang lain, kecuali dicantumkan sumbernya sesuai dengan ketentuan yang berlaku.

Demikian surat pernyataan ini saya buat, dan apabila di kemudian hari ada pelanggaran atas surat pernyataan ini, saya bersedia menerima sanksinya.

Malang, Oktober 2012

Yang membuat Pernyataan,

  
Rony Hamka.  
NIM :05.12.669

# **Implementasi Algoritma Contour Analys Untuk Pengenalan Objek Alfanumerik Menggunakan C#**

**Rony Hamka**

**Program studi Tknik Elektro S-1  
Kosentrasi T. Komputer dan Informatika  
Konsentrasi Teknik Industri, dan Instiyut Teknologi Nasional Malang  
Email : [ronyhamka@yahoo.com](mailto:ronyhamka@yahoo.com)**

**Dosen pembimbing: 1.Irmalia Suryani Faradisa,ST,MT**

**2.Sandy Natali Manja,Skom**

## **Abstrak**

Seiring dengan semakin meningkatnya kebutuhan akan digitalisasi dokumen saat ini, untuk itu diperlukan suatu *tools* yang dapat membantu pengenalan dokumen dalam bentuk digital. Aplikasi ini biasa dikenal dengan *OCR (Optical Character Recognition)*. Pembuatan aplikasi ini tidak lepas dari metode Pengolahan citra digital untuk dapat memperoleh informasi yang terkandung dalam citra sehingga dapat ditentukan polanya (*Patern Recognition*). Setelah pola ditentukan, dapat dengan mudah dilakukan analisa terhadap bentuk karakter yang ingin dikenali. Analisa ini dilakukan dengan memanfaatkan teori *Contour Analys*. Teori ini dapat mengekstrak garis terluar dari suatu karakter alfanumerik menjadi suatu kontur yang nantinya di cocokkan dengan *template*, sehingga objek tersebut dapat dikenali oleh aplikasi yang dibuat. Aplikasi ini dibangun menggunakan bahasa pemograman *C#* ditambah *library Emgu.CV* dan *library ContourAnalys* karena memiliki banyak fasilitas pengolahan citra yang dapat dimanfaatkan.

Kata kunci : *Paternrecognition, OCR, Normalized Contour Analysis*

## KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa, yang telah memberikan berkahnya, sehingga penulis dapat menyelesaikan Skripsi ini dengan baik dan lancar.

Laporan Skripsi ini merupakan salah satu persyaratan akademik dalam menyelesaikan program S1 Program Studi Teknik Elektro, Konsentrasi Komputer & Informatika, Institut Teknologi Nasional Malang. Adapun judul laporan Skripsi ini adalah:

### **“Implementasi Algoritma Contour Analys Untuk Pengenalan Objek Alfanumerik Menggunakan C#”**

Selanjutnya pada kesempatan ini penulis juga menyampaikan rasa trimakasi yang sebesar-besarnya kepada pihak yang telah banyak membantu penulis selama penyusunan tugas akhir, diantaranya:

1. Bapak Ir. Soeparno Djiwo, MT selaku Rektor Institut Teknologi Nasional Malang
  2. Bapak Ir. Sidik Nortjahjono, MT selaku Dekan Fakultas Teknologi Nasional Malang
  3. Bapak Ir. Yusuf Ismail Nahkoda, MT selaku Ketua Program Studi
  4. Bapak Dr. Aryuanto Soetedjo, ST MT selaku Sketaris Program Studi Teknik Elektro S-1 ITN Malang dan pengusul serta penyediaan ruang skripsi
  5. Ibu Irmalia Suryani F, ST MT selaku dosen pembimbing I
  6. Ibu Sandy Natali Manja, Skom selaku dosen pembimbing II
  7. Bapak Ir. Yusuf Ismail Nahkoda, MT selaku Dosen wali.
  8. Kedua orang tua, kak mil dan abang amad serta adekq yang telah memberikan dukungan untuk selalu berdoa, berusaha dan nasehat yang telah sampai saat ini.
  9. orang yang ter istimewa: Retno Astrin, yang selallu memberikan motivasi dan memberikan semangat tidak pernah henti-hentinya, makasi ya sayang I love you forever.
  10. Semua teman-teman yang selalu memberikan semangat.
  11. Semua pihak yang telah membantu penulis dalam menyelesaikan skripsi ini yang tidak bisa penulis sebutkan satu persatu.
-

Penulis berharap agar buku Skripsi ini dapat memberikan banyak manfaat bagi semua pihak yang membutuhkan, khususnya bagi rekan-rekan mahasiswa. Penulis menyadari bahwa dalam penyusunan laporan ini masih banyak kekurangan, oleh karena itu mohon maaf apabila dalam buku ini terdapat hal-hal yang kurang berkenan dihati para pembaca.

Penulis juga berharap koreksi serta saran-saran yang bermanfaat demi kesempurnaan buku Laporan ini.

Malang, Agustus 2012

Penulis

---

## Daftar Isi

Abstrak .....	iii
Kata Pengantar .....	iv
Daftar Isi.....	iv
Daftar Gambar.....	vi
Bab IPENDAHULUAN.....	1
1.1 Latar belakang .....	1
1.2 Rumusan Masalah .....	3
1.3 Tujuan.....	3
1.4 Batasan Masalah.....	3
1.5 Metodologi .....	3
1.5.1 Metode Pengumpulan Data.....	3
1.5.2 Metode Pengembangan Sistem.....	4
1.6 Sistematika Penulisan.....	5
Bab IIDasar Teori.....	7
2.1 Computer Vision .....	7
2.2 Definisi Citra, Pengolahan Citra, dan Pengenalan Pola .....	7
2.3 Pengenalan OCR .....	8
2.4 Contour Analysis.....	9
2.4.1 Konsep Utama Algoritma Contour Analysis.....	9
2.4.2 Properti Kontur .....	11
2.4.3 Produk Skalar Kontur .....	11
2.4.4 Fungsi Korelasi Kontur.....	15
2.5 Microsoft Net Framework.....	18
2.6 C# .....	19
2.7 Emgu CV.....	19
BAB IIIAnalisa Dan Perancangan Sistem .....	20
3.1 Deklarasi Umum.....	20
3.2 Analisa Kebutuhan Sistem .....	20
3.2.1 Analisa Fungsional.....	20
3.2.2 Use Case Diagram.....	21
3.2.3 Performasi OCR Kontur.....	22
3.2.4 Diagram Alir (Flowchart) .....	23
3.3 Perancangan OCR .....	24
3.3.1 Diagram Kelas.....	24

3.3.1	Diagram Kelas.....	24
3.3.2	Perancangan Image Operation .....	24
3.3.3	Pengolahan Citra pada Image Operation.....	25
3.3.4	Menambahkan Library ConturAnalys pada Aplikasi OCR Kontur	28
3.3.5	Perancangan Antar Muka ( <i>Interface</i> ).....	30
BAB IV Implementasi Dan Pengujian Sistem .....		32
4.1	Spesifikasi Perangkat Keras Dan Lunak .....	32
4.1.1	Spesifikasi Perangkat Keras.....	32
4.1.2	Spesifikasi Perangkat Lunak .....	33
4.2	Implementasi Aplikasi OCR Kontur .....	33
4.3	Pengujian Sistem .....	34
BAB V Kesimpulan .....		37
5.1	Kesimpulan.....	37
5.2	Saran .....	37
<i>Daftar Pustaka</i> .....		38
Lampiran .....		39



## Daftar Gambar

GAMBAR 1 HASIL OCR DARI TEXT CETAK YANG DI SCAN .....	8
GAMBAR 2 PENGKODEAN PADA CONTIUR ANALYS .....	10
GAMBAR 3 PROPERTI NORMALISASI PRODUK SCALAR KONTUR .....	14
GAMBAR 4 DIAGRAM USE CASE .....	22
GAMBAR 5 DIAGRAM ALIR RANCANGAN SISTEM .....	24
GAMBAR 6 DIAGRAM ALIR KELAS IMAGE OPERATION .....	25
GAMBAR 7 RANCANGAN INTERFACE .....	30
GAMBAR 8 TAMPILAN APLIKASI OCR KONTUR .....	33
GAMBAR 9 PUNGUJIAN CITRA DIGITAL PADA OBJEK SEDERHANA ..	34
GAMBAR 10 PUNGUJIAN CITRA DIGITAL PADA OBJEK KOMPLEKS ...	35
GAMBAR 11 HASIL PENGUJIAN TEKS CETAK REALTIME MODE .....	36

## BAB I

### PENDAHULUAN

#### 1.1 Latar belakang

Teknologi Informasi (TI) turut berkembang sejalan dengan perkembangan peradaban manusia. Perkembangan teknologi informasi meliputi perkembangan infrastruktur TI, seperti hardware, software, teknologi penyimpanan data (storage), dan teknologi informasi. Perkembangan teknologi tidak hanya mempengaruhi dunia bisnis, tetapi juga bidang-bidang lain, seperti kesehatan, pendidikan, pemerintahan, dan lain-lain.

Kemajuan teknologi informasi juga berpengaruh signifikan terhadap peningkatan efektivitas dan kemudahan dalam kehidupan manusia. Berbagai macam kegiatan yang membutuhkan faktor manusia dapat digantikan perannya oleh perangkat lunak yang cerdas. Tidak terkecuali pengenalan benda di kehidupan nyata, seperti pengenalan karakter alfanumerik baik berupa tulisan tangan, karakter huruf arab, karakter huruf kanji, dan lain-lain.

Adanya sistem pengenalan huruf yang cerdas akan sangat membantu usaha besar-besaran yang saat ini dilakukan banyak pihak yakni usaha digitalisasi informasi dan pengetahuan, misalnya dalam pembuatan koleksi pustaka digital, koleksi sastra kuno digital, dll.

Teknik pengenalan karakter ini sering disebut secara umum sebagai teknologi OCR (*Optical Character Recognition*). Teknologi ini bukanlah hal baru dalam ruang lingkup teknologi informasi. Teknologi OCR ini banyak ditawarkan dalam produk-produk *scanner* pada masa terkini. Selain pada produk *scanner*, teknologi OCR juga terdapat pada *Handphone*, *Smart Phone*, dan PDA (*Personal*

*Digital Assistant*) yang sudah mengimplementasikan teknologi layar sentuh dan memiliki fitur *handwritingrecognition*.

Pada proses pengenalan karakter, system mencoba mengenali apakah citra masukan yang diterima cocok dengan salah satu citra yang telah ditentukan. Keakuratan system dinilai berdasarkan ketepatan dalam mengenali karakter yang diinginkan.

Sampai dengan saat ini system OCR terus dikembangkan dengan menggunakan berbagai teori dan pendekatan guna memperoleh ketepatan dalam pengenalan karakter yang maksimal. Tapi hal ini tidak luput dari beberapa kelemahan yang kerap kali ditemukan pada pengenalan suatu Misalnya salah membaca karakter ("i" ditransfer menjadi "l" atau "t" menjadi "1"), bahkan tak jarang teks hasil scan memiliki intensitas yang sangat kecil sehingga sulit untuk dibaca.

untuk itu pada penelitian kali ini, penulis mencoba untuk mengembangkan sistem OCR dengan memanfaatkan sebuah metode atau teknik yang digunakan untuk pengenalan pola yaitu Metode Contour Analysis. Sebuah teori pengenalan pola atau *patern recognition* yang ditemukan oleh JA Fuhrman berkebangsaan Rusia ini mendeskripsikan, menyimpan, membandingkan dan menemukan obyek yang disajikan seperti karakter dalam bentuk outline eksterior atau disebut kontur. Kontur itu sendiri mengandung banyak informasi yang dapat digunakan pada proses pengenalan suatu objek. (Fuhrman JA, Introduction to the contour analysis).

Dengan memanfaatkan pengenalan pola berdasarkan kontur dari sebuah karakter diharapkan dapat membuat system OCR lebih akurat.

## 1.2 Rumusan Masalah

Berdasarkan uraian pada latar belakang maka didapat permasalahan yaitu Bagaimana mengoptimalkan akurasi pada system OCR dengan membangun sebuah aplikasi pengembangan menggunakan algoritma *Contour Analysis*, yang dapat membantu mengatasi kelemahan dalam hal mengidentifikasi teks.

## 1.3 Tujuan

Tujuan yang ingin dicapai dalam pelaksanaan tugas akhir ini adalah membangun aplikasi pengembangan sistem pengenalan teks dengan tingkat akurasi maksimal yaitu dapat mengenali hampir seluruh karakter yang terdapat pada dokumen dengan memanfaatkan algoritma *Contour Analysis*.

## 1.4 Batasan Masalah

Agar pembahasan masalah tidak menyimpang dari tujuan penelitian, maka berikut adalah beberapa batasan yang perlu dibuat yaitu :

- a. Sistem ini menangani pengenalan berdasarkan karakter alfanumerik (huruf A sampai Z, huruf a sampai z, 0 sampai 9, dan 28 simbol).
- b. Keluaran sistem berupa, Tampilan hasil pengenalan objek alfanumerik dari object hasil pemindaian, maupun secara realtime menggunakan kamera Webcam.
- c. Sistem ini dirancang menggunakan bahasa pemograman C#.

## 1.5 Metodologi

### 1.5.1 Metode Pengumpulan Data

Metode yang digunakan untuk mengumpulkan data sebagai berikut :

#### 1. Studi Lapangan

Dengan metode ini data-data diperoleh langsung dari sumber yang bersangkutan, yang dilakukan dengan cara :

a. Survey

Teknik pengumpulan data dengan cara terjun secara langsung dan mencatat secara sistematis terhadap obyek masalah.

b. Wawancara / Interview

Teknik pengumpulan data dengan mengadakan komunikasi

c. Studi Pustaka / Literatur

Pengumpulan data ini dilakukan dengan cara mencari bahan-bahan kepustakaan sebagai landasan teori yang ada hubungannya dengan permasalahan yang dijadikan obyek penelitian.

### **1.5.2 Metode Pengembangan Sistem**

Metode yang digunakan untuk pengembangan Sistem Informasi Manajemen ini adalah metodologi *waterfall* atau siklus hidup perangkat lunak. Tahap-tahap pada metodologi ini adalah

1. *Analisis dan definisi persyaratan.* Pelayanan, batasan, dan tujuan sistem ditentukan melalui konsultasi dengan user sistem. *Persyaratan ini kemudian didefinisikan secara rinci dan berfungsi sebagai spesifikasi sistem.*
2. *Perancangan sistem dan perangkat lunak.* Proses perancangan sistem membagi persyaratan dalam sistem perangkat keras atau perangkat lunak. Kegiatan ini menentukan arsitektur sistem secara keseluruhan. Perancangan perangkat lunak melibatkan identifikasi dan deskripsi abstraksi sistem perangkat lunak yang mendasar dan hubungan-hubungannya.
3. *Implementasi dan pengujian unit.* Pada tahap ini, perancangan perangkat lunak direalisasikan sebagai serangkaian program atau

unit program. Pengujian unit melibatkan verifikasi bahwa setiap unit telah memenuhi spesifikasinya.

4. *Integrasi dan pengujian sistem.* Unit program atau program individual diintegrasikan dan diuji sebagai sistem yang lengkap untuk menjamin bahwa persyaratan sistem telah terpenuhi. Setelah pengujian sistem, perangkat lunak dikirim kepada perusahaan.

## 1.6 Sistematika Penulisan

Sistematika penulisan yang digunakan dalam penyusunan skripsi ini adalah sebagai berikut:

### BAB I : Pendahuluan

Bab ini berisikan tentang Latar Belakang, Maksud dan Tujuan, Tempat dan Waktu Pelaksanaan, Ruang Lingkup, Metode Pengumpulan Data, dan Sistematika Penulisan.

### BAB II : Tinjauan Pustaka

Bab ini berisikan tentang pengolahan citra dan pengenalan OCR, teori Microsoft .NET Framework, dan C#

### BAB III : Metodologi

Berisi gambaran umum perusahaan berupa analisa sistem yang dibutuhkan dan perencanaan objek yang diuji atau desain sistem yang akan dikembangkan menurut metode penelitian yang dilakukan.

### BAB IV : Hasil dan Pembahasan Sistem Informasi

Bab ini menyajikan pembuatan objek uji, hasil pengujian serta pembahasan dari hasil pengujian yang dilakukan.

## BAB V : Penutup

Bab ini berisi kesimpulan dan saran-saran dari hasil pembahasan skripsi ini.

## Bab II

### Dasar Teori

#### 2.1 Computer Vision

Menurut Michael G. Fairhurst (1988) computer vision merupakan ilmu yang mempelajari cara komputer dapat mengenali objek yang diamati. *Computer vision* merupakan kombinasi antara pengolahan citra dan pengenalan pola yang bersama intelegensia semu akan mampu menghasilkan sistem intelegen visual (*Visual Intelligence Systems*).

#### 2.2 Definisi Citra, Pengolahan Citra, dan Pengenalan Pola

Citra adalah representasi dari suatu objek nyata baik dalam bentuk dua dimensi maupun tiga dimensi menjadi bentuk gambar digital yang dimengerti oleh komputer ( Anil K. Jain, 1989). Definisi citra lainnya adalah sebuah fungsi identitas warna dua dimensi  $f(x,y)$  dimana  $x$  dan  $y$  mewakili koordinat lokasi suatu titik dan nilai dari fungsi yang merupakan tingkat intensitas warna atau tingkat keabu-abuan dari titik tersebut (Robert J. Schalkoff, 1989).

Pengolahan citra (*image Processing*) merupakan bidang yang berhubungan dengan transformasi citra (*image*) yang bertujuan untuk mendapatkan kualitas citra yang baik ataupun memanfaatkan informasi citra tersebut untuk berbagai implementasi yang bermanfaat. (Michael G. Fairhurst, 1988).

Pengenalan Pola (*Pattern Recognition*) merupakan bidang yang berhubungan dengan proses identifikasi obyek pada citra atau interpretasi citra. Proses ini bertujuan untuk mengekstrak informasi atau pesan yang disampaikan oleh gambar atau citra. (Michael G. Fairhurst, 1988).



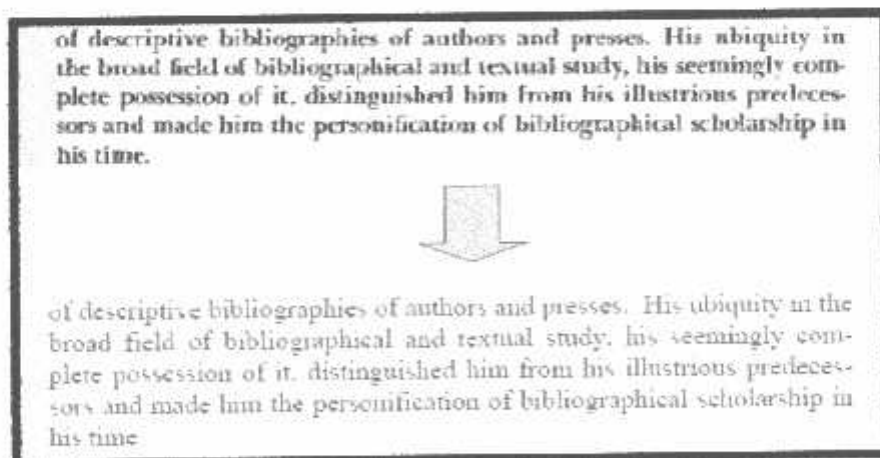
### 2.3 Pengenalan OCR

*Optical character recognition* atau biasa disingkat *OCR* adalah sebuah teknik yang dapat membantu komputer dalam menerjemahkan sebuah gambar berisi teks, baik itu berasal dari tulisan tangan, mesin ketik, maupun hasil cetakan komputer.

OCR adalah sebuah teknologi lama yang muncul pada tahun 1929 yang dikenalkan oleh penciptanya yaitu Gustav Tauschek. Kemudian didaftarkan hak patennya di Jerman. ([http://en.wikipedia.org/wiki/Optical\\_character\\_recognition](http://en.wikipedia.org/wiki/Optical_character_recognition)).

Dalam proses OCR, ada beberapa komponen yang bekerja dalam proses mengenali karakter teks. Komponen-komponen ini meliputi identifikasi pola karakter teks, artificial intelligence, dan Macine Vision. Seperti yang disebutkan sebelumnya, teknologi ini juga berkembang, tujuannya tentu agar pengenalan teks digambar menjadi lebih cepat dan akurat. Selama ini teknologi OCR menggunakan dua metode, yaitu *Matric Matching* dan *Feature Extraction*. Dimana kedua metode tersebut memiliki kelebihan dan kekurangan masing-masing pada proses implementasinya.

Proses OCR melibatkan beberapa langkah termasuk segmentasi, ekstraksi fitur, dan klasifikasi. Contoh proses pengenalan karakter alfanumerik seperti



*Gambar 1 Hasil OCR dari text cetak yang di scan*

## 2.4 Contour Analysis

*Contour Analysis (CA)* memungkinkan untuk menggambarkan, menyimpan, membandingkan dan menemukan objek yang disajikan dalam bentuk garis-garis eksterior-kontur.

Hal ini diduga bahwa kontur berisi informasi yang diperlukan pada bentuk objek. Pada kasus Point Interior objek tidak dibahas dan digunakan pada proses. Penerapan algoritma CA yaitu terbatas pada pengekstrakan tampilan gambar pada ruang 2-dimensi.

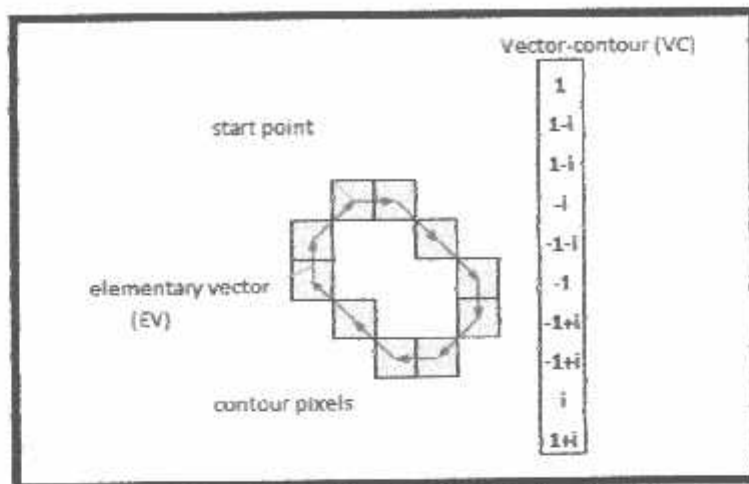
Penggunaan Contour Analysis memungkinkan dapat menyelesaikan masalah pada pengenalan pola (Pattern Recognition), mengubah posisi serta mengubah skala gambar objek yang ingin di olah.

### 2.4.1 Konsep Utama Algoritma Contour Analysis

Pada awalnya, kita mendefinisikan seperti kontur objek. Kontur merupakan batas dari objek, populasi titik (pixel), memisahkan objek dari latar belakang.

Dalam sistem *Computer Vision*, beberapa format pengkodean kontur digunakan pengkodean pada teori Freeman, pengkodean dua-dimensi, dan yang paling dikenal pengkodean poligonal. Tapi semua format pengkodean yang tidak digunakan dalam CA.

Sebaliknya, dalam CA kontur dikodekan oleh urutan yang terdiri dari bilangan kompleks. Pada kontur, titik yang disebut sebagai titik awal adalah tetap. Kemudian, kontur dipindai (searah jarum jam), dan masing-masing vektor offset dicatat oleh bilangan kompleks  $a + ib$ . Dimana  $a$  mewakili sumbu x, dan  $b$  mewakili pada sumbu y.



Gambar 2 Pengkodean Pada Contour Analys

Vektor terakhir dari kontur selalu mengarah ke titik awal. Setiap vektor pada kontur dinamakan *Elementary Vector (EV)*. Dan setiap urutan vektor berisi nilai yang disebut *Vector Contour (VC)*.

Vector-contour akan kita tandai sebagai huruf Yunani besar dan elementary vector huruf Yunani kecil.

Sehingga, vector contour  $\Gamma$  dengan panjang  $k$  dapat ditulis sebagai berikut:

$$\Gamma = (\gamma_0, \gamma_1, \dots, \gamma_{k-1})$$

Mengapa dalam CA sebuah koding bernilai-kompleks digunakan? Karena operasional atas sebuah kontur sama seperti atas sebuah vektor dari angka-angka yang kompleks akan memiliki properti matematika yang mengesankan, dibandingkan dengan mode koding lainnya.

Pada dasarnya, koding yang kompleks akan dekat dengan koding dua-dimensi dimana kontur didefinisikan sebagai sebuah populasi EV yang diberikan dalam koordinat-koordinat dua-dimensi. Tapi sebuah perbedaan antara operasi produk skalar untuk vektor dan untuk angka-angka yang kompleks – akan sangat bervariasi. Situasi ini juga memberikan prioritas bagi metode CA.

### 2.4.2 Properti Kontur

1. Jumlah sebuah EV dari sebuah kontur tertutup akan sama dengan nol. Sangatlah penting – ketika elementary vector dihasilkan di titik awal, jumlah mereka akan sama dengan sebuah vektor-nol.
2. contour-vector tidak tergantung pada transposisi paralel dari imej sumber. Ketika kontur dikodekan relatif terhadap titik awal, mode koding ini akan invarian terhadap perubahan dalam kontur inisial.
3. Pemutaran imej pada sudut tertentu akan sama dengan pemutaran masing-masing EV dari sebuah kontur dengan sudut yang sama.
4. Modifikasi titik awal berlaku untuk perubahan siklus VC. Ketika EV dikodekan terkait dengan titik sebelumnya, sudah jelas bahwa sebuah modifikasi dalam titik awal, sekuensi EV akan sama, tapi EV pertama adalah apa yang dimulai di titik awal.
5. Penskalaan ulang imej sumber dapat dianggap sebagai perkalian masing-masing EV dari sebuah kontur terhadap faktor skala.

### 2.4.3 Produk Skalar Kontur

Sebagai produk skalar kontur,  $\Gamma$  dan  $N$  membutuhkan angka-angka yang kompleks:

$$\eta = (\Gamma, N) = \sum_{n=0}^{k-1} (\gamma_n, v_n) \quad (1)$$

Dimana  $k$  – dimensionalitas dari sebuah VC,  $\gamma_n$  – elementary vector ke- $n$  dari kontur  $\Gamma$ ,  $v_n$  – EV ke- $n$  dari kontur  $N$ .  $(\gamma_n, v_n)$  – produk skalar dari angka-angka kompleks yang dihitung sebagai berikut:

$$(a+ib, c+id) = (a+ib)(c-id) = ac + bd + i(bc-ad) \quad (2)$$

Mari kita perhatikan bahwa dalam sebuah CA maka produk skalar hanyalah sebuah VC dari dimensionalitas yang identik yang dianggap. Yakni angka elementary vector dalam kontur harus serupa.

Produk skalar dari vektor biasa dan produk skalar dari angka-angka kompleks akan berbeda. Jika kita mengalikan sebuah EV hanya sebagai sebuah vektor maka produk skalar mereka akan tampak seperti:

$$((a, b), (c, d)) = ac + bd \quad (3)$$

Bandingkan hasil ini dengan rumus (2) maka kau akan mendapatkan:

- Hasil produk skalar vektor adalah angka yang riil. Dan hasil dari produk angka yang rumit – sebuah angka yang kompleks.
- Bagian riil dari produk skalar angka-angka kompleks akan serupa dengan produk skalar vektor yang bersangkutan. Sehingga produk kompleks akan memasukkan produk skalar vektorial.

Kini coba kita ingat aljabar linier. Lebih tepatnya – pengertian fisik dan properti produk skalar. Produk skalar akan sama dalam aljabar linier sebagai produk panjang vektor dalam sebuah kosinus sudut diantaranya. Hal ini berarti bahwa dua vektor yang saling tegak lurus akan selalu memiliki produk skalar nol, kolinier sebuah vektor – sebaliknya, akan memberikan nilai maksimal dalam produk skalar.

Properti-properti produk ini membuatnya dapat digunakan sebagai sebuah pengukur kedekatan vektor yang tepat. Jika lebih dari ini – semakin sedikit sudut antara vektor, “semakin dekat” mereka satu sama lain. Untuk vektor yang saling tegak lurus – nilainya akan berkurang mendekati nol, dan kemudian akan bernilai negatif untuk vektor-vektor yang diarahkan berlawanan. Tampaknya, produk skalar (1) juga memiliki properti-properti yang sama.

Mari saya perkenalkan satu konsep lain – normalized scalar product (NSP):

$$\eta = \frac{(\Gamma, N)}{|\Gamma||N|} \quad (4)$$

Dimana  $|\Gamma|$  dan  $|N|$  – norma (panjang) kontur akan dihitung sebagai berikut:

$$|\Gamma| = \left( \sum_{n=0}^{k-1} |\gamma_n|^2 \right)^{\frac{1}{2}} \quad (5)$$

NSP dalam ruang angka-angka kompleks, juga merupakan sebuah angka-angka kompleks (complex numbers).

Sehingga, satu adalah nilai norma yang paling memungkinkan dalam NSP (hal ini berasal dari sebuah pertidaksamaan Cauchy-Bunyakovsky-Schwarz:  $|ab| \leq |a||b|$ ), dan akan tercapai hanya jika

$$\Gamma = \mu N \quad (6)$$

dimana  $\mu$  - adalah angka kompleks arbitrary.






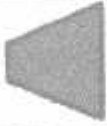

Bagaimana cara mengimplementasikannya? Kita dapat mengingat pengertian fisik dari perkalian angka-angka kompleks. Pada perkalian angka-angka kompleks, panjang mereka akan dikali, dan argumen (sudut) – akan dijumlahkan. Kontur  $\mu N$  berarti memiliki kontur yang sama seperti  $N$ , tapi diputar pada kedua skalanya. Skala dan pemutaran (turn) didefinisikan oleh angka kompleks  $\mu$ .

Jadi, aturan NSP mencapai nilai maksimum – satu, hanya terjadi jika kontur  $\Gamma$  adalah kontur yang sama seperti  $N$ , tapi diputar pada beberapa sudut dan diskalakan pada koefisien tertentu.

Untuk sebuah contohnya, kami menganjurkan sebuah perkalian skalar dari sebuah kontur terhadap dirinya sendiri, tapi diputar pada sebuah sudut tertentu.

Jadi jika akan menghitung sebuah NSP dari sebuah vektor terhadap dirinya sendiri, kita menerima  $NSP=1$ , jika akan memutar sebuah kontur pada 90 derajat,

kita akan mendapatkan  $NSP=0+i$ , diputar pada 180 derajat akan memberikan nilai  $NSP=-1$ . Sehingga, bagian yang riil dari NSP akan memberikan kita sebuah kosinus sudut antar kontur, dan norma/aturan NSP akan selalu sama dengan 1.

	NSP	$Re(NSP)=\cos(a)$	$ NSP $
 x	1	1	1
 x 	i	0	1
 x 	-1	-1	1
 x 	-i	0	1

Gambar 3 Properti normalisasi produk scalar kontur

Serupa dengan ini, jika kita meningkatkan VC pada sejumlah koefisien riil tertentu (skala) maka kita juga akan menerima  $NSP = 1$  (hal ini dapat terlihat dari rumus (4)).

**Catatan:** Norma/aturan NSP akan invarian terhadap transposisi, rotasi dan penskalaan kontur.

Jadi, norma atau aturan NSP kontur memberikan nilai satu hanya dalam peristiwa dimana dua kontur akan setara dalam hal pemutaran dan skala. Selain itu, norm NSP akan kurang dari satu.

Hal ini merupakan kesimpulan penting dari sebuah CA. Sebenarnya, norma/aturan dari sebuah NSP adalah sebuah invarian terhadap transposisi, rotasi dan penskalaan kontur. Jika terdapat dua kontur yang identik maka NSP mereka akan selalu memberikan nilai satu, tidak tergantung pada dimana kontur tersebut berada, atau pada sudut rotasi dan skala mereka. Serupa dengan ini, jika kontur

bervariasi NSP mereka akan kurang dari 1, dan tidak tergantung pada tempat, rotasi dan skala kontur tersebut.

**Catatan:** Norma/aturan NSP diukur dari kedekatan kontur.

Norma/aturan memberikan sebuah pengukuran likeness kontur, dan argumen sebuah NSP (sama dengan atan (b/a)) – memberikan kita sebuah sudut rotasi kontur terhadap satu sama lain.

#### 2.4.4 Fungsi Korelasi Kontur

Di dalam subbab sebelumnya, kita memastikan bahwa NSP adalah rumus yang sangat berguna untuk mencari kontur yang serupa diantara mereka sendiri. Sayangnya, terdapat satu situasi yang tidak mengizinkan penggunaan NSP secara langsung. Dan situasi ini adalah – sebuah pilihan titik awal.

Masalahnya adalah persamaan (6) hanya akan dapat dicapai jika titik awal kontur – coincide (serupa). Jika kontur identik, tapi referensi EV dimulai dengan titik awal yang berbeda maka norma NSP dari kontur seperti ini tidak akan sama dengan satu.

Mari kita lihat konsep *fungsi interkorelasi* (ICF) dari dua kontur:

$$\tau(m) = (\Gamma, N^{(m)}), m=0, \dots, k-1 \quad (7)$$

Dimana  $N^{(m)}$  – sebuah kontur yang diterima dari  $N$  oleh perubahan siklus atas EV-nya dalam elemen  $m$ .

Contohnya, jika  $N^{(m)}=(n1, n2, n3, n4)$ ,  $N^{(1)}=(n2, n3, n4, n1)$ ,  $N^{(2)}=(n3, n4, n1, n2)$  dan seterusnya.

Nilai dari fungsi ini menunjukkan bahwa kontur  $\Gamma$  dan  $N$  sangat mirip jika mengubah titik awal  $N$  ke posisi  $m$ . ICF didefinisikan dalam semua set angka integral tapi ketika siklus mengalami perubahan dalam  $k$  maka akan mengarahkan kita pada sebuah kontur inisial ICF yang periodik, dengan fase  $k$ . Oleh karenanya,



kita akan menarik nilai fungsi ini hanya dalam limit antara 0 hingga k-1. Mari kita ungkapkan besarnya memiliki norma maksimum diantara nilai-nilai sebuah ICF:

$$\tau_{\max} = \max\left(\frac{\tau(m)}{|\Gamma||N|}\right), m=0, \dots, k-1 \quad (8)$$

Dari determinasi sebuah NSP dan ICF, sudah tampak jelas bahwa  $\tau_{\max}$  adalah sebuah pengukuran kemiripan atas dua kontur, invarian terhadap transposisi, penskalaan, rotasi dan perubahan titik awal.

Sehingga, norma  $|\tau_{\max}|$  akan menunjukkan level kemiripan kontur, dan mencapai nilai satu untuk kontur identik, dan argumen  $\arg(\tau_{\max})$  akan memberikan sebuah sudut rotasi dari satu kontur, relatif terhadap yang lainnya.

**Catatan:** Nilai maksimum norma ICF adalah sebuah pengukuran kesamaan dari dua kontur. Danakan invarian terhadap transposisi, penskalaan, rotasi dan perubahan titik awal.

Konsep lain yaitu *fungsi autokorelasi* (ACF). Fungsi ACF ini adalah sebuah ICF dimana  $N=\Gamma$ . Bahkan fungsi ini merupakan sebuah produk skalar sebuah kontur atas dirinya sendiri dalam beragam perubahan titik awal:

$$v(m) = (\Gamma, \Gamma^{(m)}), m=0, \dots, k-1 \quad (9)$$

Mari kita lihat beberapa properti ACF:

1. ACF tidak tergantung pada pemilihan titik awal sebuah kontur. Sebenarnya, kita melihat penentuan sebuah produk skalar (1). Seperti yang kita lihat, modifikasi titik awal akan mengarah pada sebuah modifikasi susunan elemen-elemen penjumlahan dan tidak menghasilkan sebuah modifikasi dalam penjumlahan. Kesimpulan ini tidak terlalu jelas tapi jika kita mempertimbangkan hal ini dalam pengertian ACF maka hal ini sudah jelas.

2. Norma sebuah ACF akan asimetris terkait sebuah referensi sentral  $k/2$ . Ketika ACF adalah total produk pasangan dari sebuah EV kontur maka masing-masing pasangan akan bertemu dua kali dalam interval dari 0 hingga  $k$ .

misalnya,  $N = (n1, n2, n3, n4)$ , kita menuliskan nilai ACF untuk  $m$  yang berbeda:

$$ACF(0) = (n1, n1) + (n2, n2) + (n3, n3) + (n4, n4)$$

$$ACF(1) = (n1, n2) + (n2, n3) + (n3, n4) + (n4, n1)$$

$$ACF(2) = (n1, n3) + (n2, n4) + (n3, n1) + (n4, n2)$$

$$ACF(3) = (n1, n4) + (n2, n1) + (n3, n2) + (n4, n3)$$

$$ACF(4) = (n1, n1) + (n2, n2) + (n3, n3) + (n4, n4)$$

Mari kita lihat bahwa item-item dalam sebuah  $ACF(1)$  akan sama, seperti dalam  $ACF(3)$ , didalam permutasi faktor. Dan mengingat hal ini untuk angka kompleks  $(a,b) = (b,a)^*$ , kita mendapatkan  $ACF(1) = ACF(3)^*$ , dimana  $*$  adalah sebuah tanda konjugasi angka kompleks.

Dan ketika  $|a^*| = |a|$  maka menyatakan bahwa norma  $ACF(1)$  dan  $ACF(3)$  akan serupa.

Sama dengan ini, norma dari  $ACF(0)$  dan  $ACF(4)$  akan serupa.

Lebih jauh lagi, dimanapun dibawah sebuah ACF kita akan memahami hanya sebuah bagian fungsi dalam interval 0 hingga  $k/2$  ketika sisa fungsi – akan simetris terhadap bagian pertamanya.

3. Jika kontur memiliki kesimetrisan apapun untuk melakukan perputaran (turn) maka ACF-nya akan memiliki kesimetrisan yang serupa. Sebagai contoh, kami membawa grafik ACF untuk beberapa kontur:

Di dalam gambar, norma ACF diwakili oleh warna biru gelap (sebuah ACF direpresentasikan hanya untuk interval dari 0 hingga  $k/2$ ).

Seperti yang kita lihat, semua kontur kecuali yang terakhir, akan memiliki simetri untuk berputar (turn) dan ACF mengarah pada kesimetrisan. Kontur terakhir dari contoh ini tidak memiliki kesimetrisan, dan dalam grafik ditunjukkan ACF-nya tidaklah simetris.

4. Dalam sebuah pengertian, akan memungkinkan untuk menganggap ACF kontur sebagai karakteristik bentuk sebuah kontur. Sehingga, bentuk, mendekati sebuah lingkaran akan memiliki nilai norma ACF yang seragam (lihat gambar sebuah lingkaran). Bentuk-bentuk yang bersudut kuat dalam satu arah – memiliki cekungan dalam di bagian tengah ACF (lihat gambar persegi). Bentuk yang dapat diputar, memiliki nilai maksimal ACF di tempatnya yang tepat (lihat ACF kuadrat).
5. Aturan atau norma ACF tidak akan tergantung pada skala, posisi, rotasi dan pemilihan titik awal sebuah kontur. Hal ini berasal dari titik pertama dan dari proporsi NSP.

(Fuhrman JA, Introduction to the contour analysis).

## 2.5 Microsoft Net Framework

Microsoft .NET Framework ( dibaca Microsoft Dot Net Framework ) adalah sebuah komponen yang dapat ditambahkan ke sistem operasi Microsoft Windows atau telah terintegrasikan ke dalam Windows (mulai Windows Server 2003 dan versi-versi Windows terbaru). Kerangka kerja ini menyediakan sejumlah besar solusi program untuk memenuhi kebutuhan-kabutuhan umum suatu program baru, dan mengatur eksekusi program-program yang ditulis secara khusus untuk framework ini..NET Framework adalah kunci penawaran utama dalam Microsoft, dan dimaksudkan untuk digunakan oleh sebagian besar aplikasi-aplikasi baru yang dibuat untuk platform windows. Dalam perkembangannya, .NET Framework seringkali dikaitkan pula dengan versi visual studio yang sesuai

dengan dukungan versi yang bersangkutan untuk pengembangan aplikasi.[wikipedia].

## 2.6 C#

C# sering dianggap sebagai penerus C++ atau versi canggih, karena ada anggapan bahwa tanda # adalah 4 tanda tambah yang disusun sedemikian rupa sehingga terbentuk tanda pagar, akan tetapi terlepas dari benar tidaknya tanggapan tersebut C# adalah sebuah bahasa pemrograman yang berorientasi pada objek yang dikembangkan oleh Microsoft dan menjadi salah satu bahasa pemrograman yang mendukung .NET programming melalui visual studio.

C# didasarkan pada bahasa pemrograman C++.C# juga memiliki kemiripan dengan beberapa bahasa pemrograman seperti Visual Basic, Java, Delphi, dan tentu saja C++. C# memiliki kemudahan syntax seperti Visual Basic,, dan tentu saja ketangguhan seperti Java dan C++. Kemiripan-kemiripan ini tentunya memudahkan programmer dari berbagai latar belakang bahasa pemrograman tidak perlu waktuyang lama untuk menguasainya, karena bagaimanapun juga C# lebih sederhana dibandingkan bahasa-bahasa pemrograman seperti C++ dan Java.

C# didesain oleh program designer dari Microsoft, Andres Hajlsberg.Sebelum bekerja pada Microsoft, Andres bekerja di Borland, tempat dia menulis Pascal Compiler. Sebelum mengembangkan C#, Andres mengetahui berbagai macam kekurangan pada bahasa C++, Delphi, Java, dan Smaltalk, karena itu Andres menciptakan bahasa C# yang lebih tangguh. Hal ini juga menjelaskan mengapa C# memiliki kemiripan dengan beberapa bahasa tersebut. (Tim Wahana Komputer, 2008).

## 2.7 Emgu CV

Emgu CV merupakan lintas *platform* yang berisi library pengolahan pengolahan gambar yang dapat dipanggil pada .NET dengan berbagai bahasa C#, VB, VC++, dll. Tidak seperti OpenCV, DotNet Emgu CV dapat berjalan di Linux, Solaris dan Mac OS X (sumber: <http://www.emgu.com>).

## BAB III

### *Analisa Dan Perancangan Sistem*

#### 3.1 Deklarasi Umum

*Dalam skripsi ini penulis akan membangun sebuah sistem aplikasi untuk pengenalan karakter alfanumerik, yang biasa dikenal dengan Optical Character Recognition (OCR) dengan memanfaatkan algoritma Analisa Kontur menggunakan bahasa pemrograman C#.*

Aplikasi yang diberi nama **OCR Kontur** merupakan aplikasi yang memudahkan komputer untuk mengenali obyek/ karakter alfanumerik dari suatu teks cetak yang diubah dalam bentuk digital dengan cara dilakukan pemindaian terhadap text cetak tersebut. Selain melakukan pengenalan karakter terhadap file teks cetak yang telah berbentuk digital baik dalam format JPEG dan Bitmap aplikasi ini juga dapat melakukan pengenalan karakter alfanumerik secara *realtime* menggunakan kamera webcam.

#### 3.2 Analisa Kebutuhan Sistem

Analisis sistem adalah penguraian dari program yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi permasalahan. Bagian analisis ini terdiri atas analisis fungsional, analisis performansi, gambaran sistem dari sudut pandang *user* yang dinyatakan dalam *use case* diagram, dan gambaran alur sistem dalam bentuk *flowchart*.

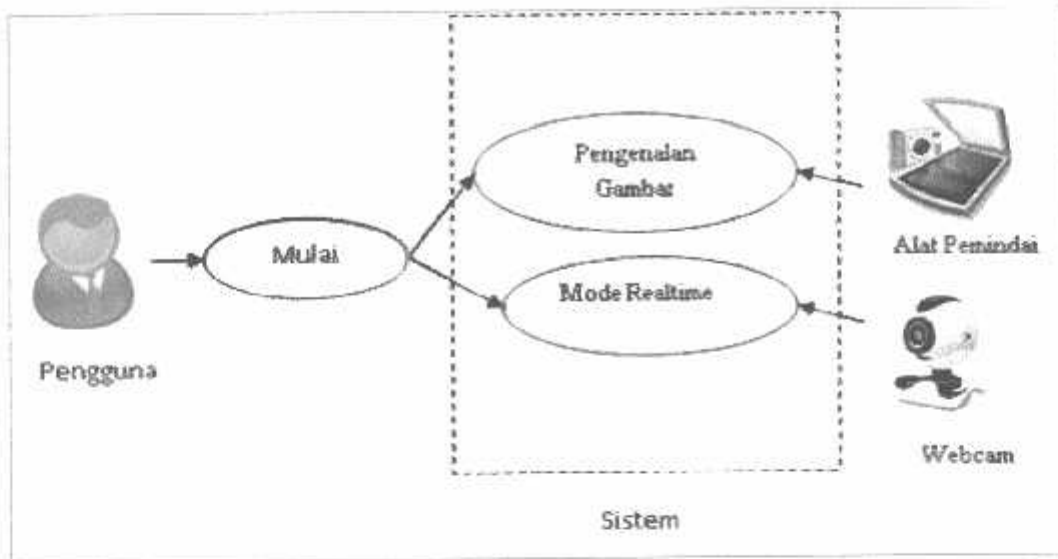
##### 3.2.1 Analisa Fungsional

Analisis fungsional merupakan paparan mengenai kemampuan yang dapat dilakukan oleh aplikasi **OCR Kontur**. Yang meliputi:

- a. Sistem ini mampu menangani pengenalan berdasarkan karakter alfanumerik (A-Z, a-z, 0-9, dan 28 simbol).
- b. Sistem ini mampu melakukan pengenalan karakter alfanumerik dari file cetak yang berbentuk *image* (JPEG dan Bitmap).
- c. Selain file *image*, system ini juga mampu mengenali karakter alfanumerik secara *realtime* menggunakan webcam.
- d. System ini melakukan proses kontur dengan membuat region of interest (ROI) berbentuk persegi dari masing-masing karakter pada sebuah gambar dalam hal ini file cetak yang telah di pindai.
- e. Aplikasi ini memungkinkan kita untuk memproses *image* dalam sebuah mode *progresif*. Hal ini berarti bahwa kita akan dapat menyeleksi kontur dalam indikasi apapun (misalnya dalam bentuk persegi atau batasan gradien, atau menurut kekontrasan, dsb). Dan kemudian memproses konturing pertama, dan memunculkan hasil. Kontur yang belum diproses akan tersimpan di dalam sistem.

### 3.2.2 Use Case Diagram

*Use case* merupakan gambaran skenario dari interaksi antara *user* dengan sistem. Sebuah diagram *use case* menggambarkan hubungan antara aktor dan kegiatan yang dapat dilakukannya terhadap aplikasi.



Gambar 4 Diagram Use Case

Pada diagram di atas terdiri dari aktor dan 2 use case. Alur ini dimulai dari ketika user memulai menjalankan program **OCR Kontur**, terdapat 2 use case yaitu *mode Open File* dan *Mode Realtime*. Pada saat aplikasi di buka, user akan otomatis berada pada *mode Realtime* dan secara otomatis sistem langsung melakukan pengenalan karakter teks cetak menggunakan fasilitas webcam yang tersedia pada perangkat keras.

Ketika pengguna memilih *use case Open File* secara otomatis sistem akan menghubungkan pengguna ke sebuah direktori penyimpanan file berupa gambar digital hasil dari pemindaian teks cetak yang ingin di kenali.

### 3.2.3 Performasi OCR Kontur

OCR Kontur merupakan aplikasi yang berjalan di lingkungan sistem operasi windows. Terdapat beberapa keterbatasan yang ditemui. Sehingga perlu diperhatikan guna menjadi acuan dalam pengembangan OCR Kontur, diantaranya:

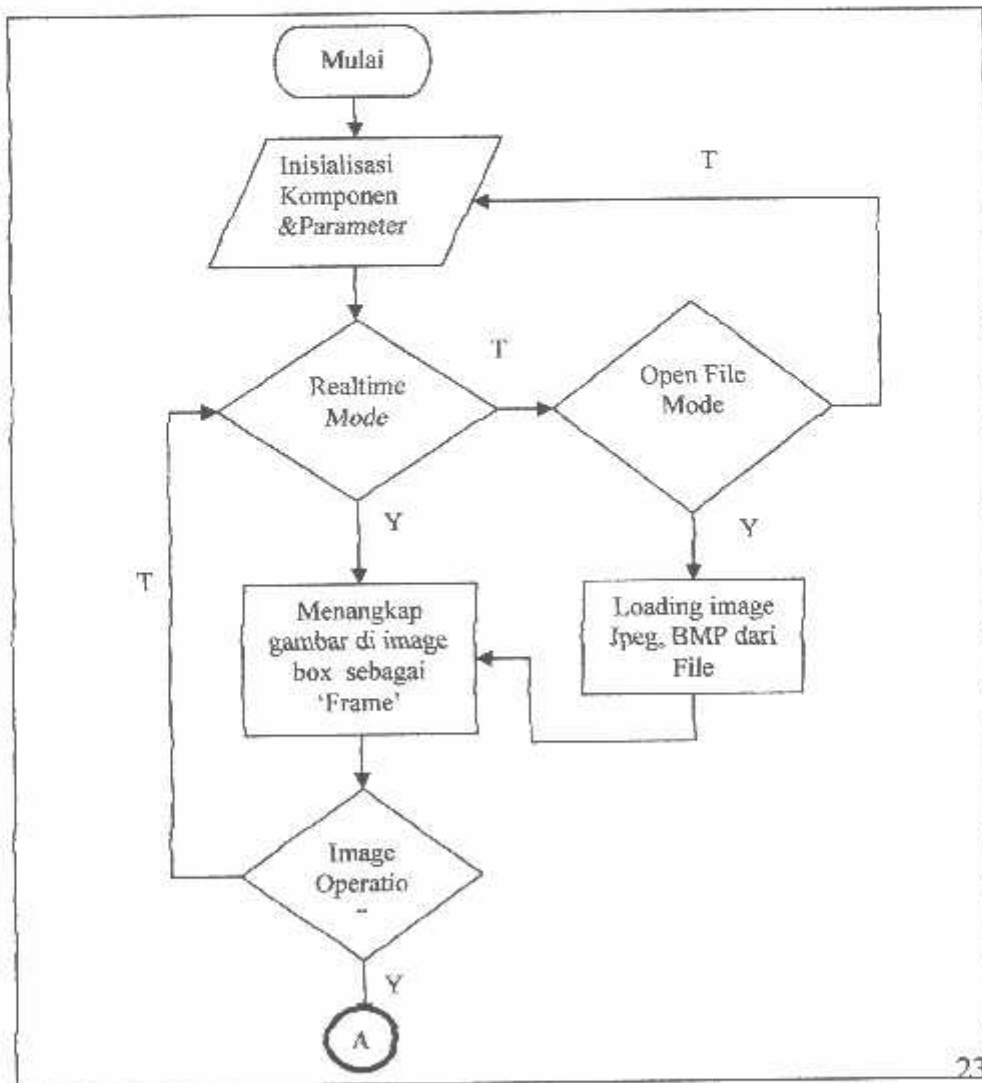
- Aplikasi ini dibuat menggunakan visual studio 2010 yang dibuat dengan lingkungan kerangka kerja .NET Framework 4. Sedangkan bawaan

windows terbaru hanya sampai .NET Framework 3.5 Sehingga untuk computer yang tidak terinstal program visual studio harus menginstal .NET Framework 4 untuk dapat menjalankan aplikasi ini.

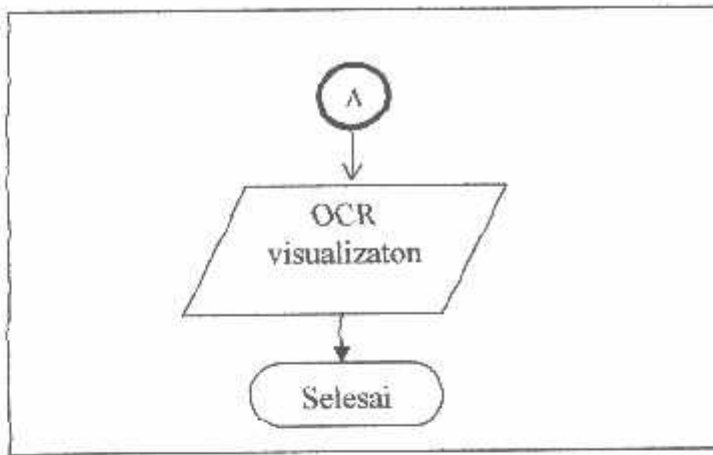
- Aplikasi ini belum dapat mengekstrak karakter menjadi sebuah file melainkan hanya tampilan pada image box. Hal ini disebabkan karena kontur diekstrak berdasarkan satu kesatuan karakter sehingga dengan menggunakan metode paint pada image box dirasa lebih tepat.

### 3.2.4 Diagram Alir (Flowchart)

Diagram Alir atau *Flowchart* merupakan serangkaian bagan-bagan yang menggambarkan alir program. Pada diagram alir ini digambarkan urutan prosedur dalam aplikasi **OCR Kontur**, berikut adalah *Flowchart* aplikasi **OCR Kontur** yang di perlihatkan pada gambar berikut.







Gambar 5 Diagram Alir Rancangan Sistem

### 3.3 Perancangan OCR

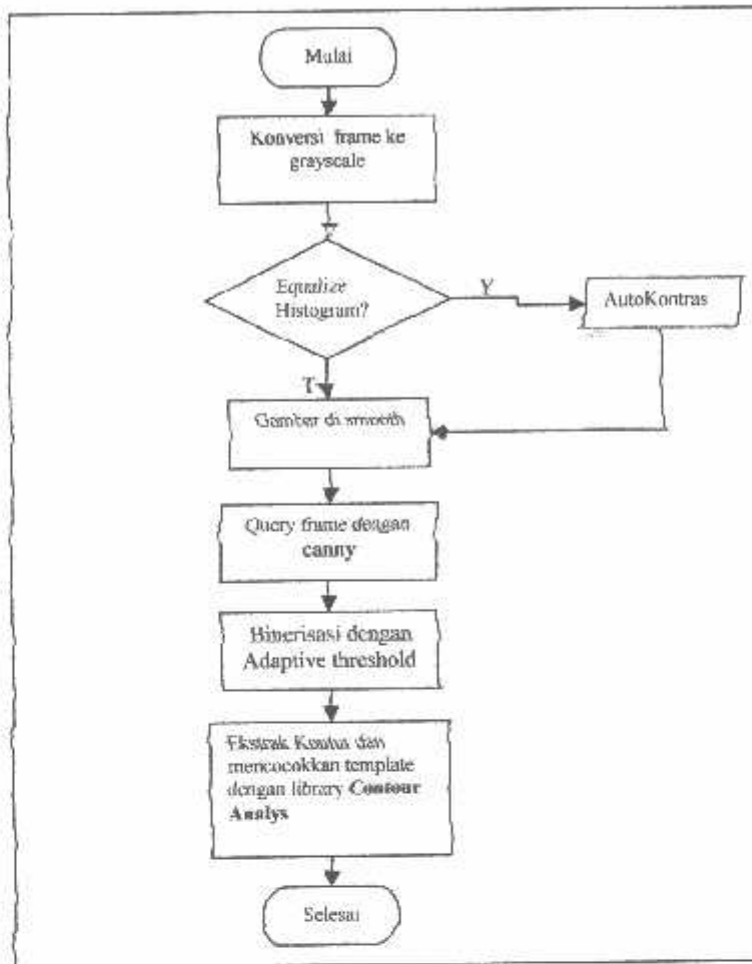
Perancangan dilakukan untuk menggambarkan, merencanakan dan membuat sketsa atau pengaturan beberapa elemen yang terpisah kedalam satu kesatuan yang utuh dan berfungsi.

#### 3.3.1 Diagram Kelas

Diagram kelas merupakan diagram struktural yang memodelkan sekumpulan kelas, *interface*, kolaborasi dan relasinya. Diagram kelas digambarkan dengan kotak, yang pada dasarnya terbagi atas tiga bagian yaitu, Nama Kelas, Atribut, dan Operasi. Diagram kelas digunakan untuk menggambarkan proses statik dari aplikasi OCR.

#### 3.3.2 Perancangan Image Operation

Kelas Image Operation merupakan kelas utama dalam pembangunan aplikasi ini, dimana kelas ini berisi proses awal pengoperasian citra dengan melakukan pengolahan citra digital dalam rangka pengenalan pola untuk di ekstrak konturnya dan di cocokkan dengan template yang ada. Berikut adalah alur perancangan kelas image operation dalam bentuk *flowchart*



Gambar 6 Diagram Alir kelas Image Operation

### 3.3.3 Pengolahan Citra pada Image Operation

Metode yang dipakai dalam pembuatan aplikasi ini tidak lepas dari peranan pengolahan citra digital. Berikut pengolahan citra yang dilakukan dalam pembuatan aplikasi ini :

1. Merubah gambar dari RGB ke grayscale, yaitu dari setiap pixel yang berisi tiga komponen warna RGB (Red Green Blue) ke satu komponen warna. Dengan skala keabuan 0-255, semakin tinggi skala ke abuan semakin putih warna pixelnya.



Implementasi pada kode C# yaitu:

```
public void ProcessImage(Image<Bgr, byte> frame)
{
    ProcessImage(frame.Convert<Gray, Byte>());
}
```

- Ubah dari grayscale ke Biner, artinya kumpulan pixel yang tadinya berkisar antara 0-255 hanya dua nilai saja yaitu 0 dan 1, dimana 0 sama dengan hitam dan 1 sama dengan putih. Hal ini dilakukan dengan menentukan batas ambang yaitu 50. Dimana untuk piksel dengan nilai intensitas di bawah 50 akan dianggap 0 dan piksel dengan intensitas diatas 50 dianggap 1. Setelah itu di terapkan metode **Adaptive Threshold** terhadap image tadi menggunakan kelas `CvInvoke.cv` pada library `Emgu.CV` tapi untuk implementasinya kita memerlukan parameter masukkan yaitu `Threshold Block Size` dan `Threshold Parameter`. Masing-masing dengan nilai 4 dan 1.2d yang di deklarasikan di setting.



Implementasi pada kode C# yaitu:

```
CvInvoke.cvAdaptiveThreshold(grayFrame, grayFrame, 255,
    Emgu.CV.CvEnum.ADAPTIVE_THRESHOLD_TYPE.CV_ADAPTIVE
    _THRESH_MEAN_C,
```

```
Emgu.CV.CvEnum.THRESH.CV_THRESH_BINARY,  
adaptiveThresholdBlockSize + adaptiveThresholdBlockSize % 2 + 1,  
adaptiveThresholdParameter);
```

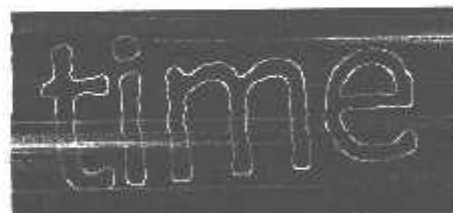
3. Ekstraksi kontur menggunakan metode CHAIN\_APPROX\_METHOD pada library Emgu.CV



Implementasi pada kode C# yaitu:

```
var sourceContours =  
grayFrame.FindContours(Emgu.CV.CvEnum.CHAIN_APPROX_METH  
OD.CV_CHAIN_APPROX_NONE,  
Emgu.CV.CvEnum.RETR_TYPE.CV_RETR_LIST)
```

4. Kemudian kontur di filter menggunakan parameter linear dan memulai operasi contour dengan library ContourAnalysis.dll.



5. Kemudian mencocokkan deskripsi bentuk kontur template dengan template yang ada dengan menggunakan fungsi FoundTemplateDesc pada library kontur ContourAnalysis.dll.

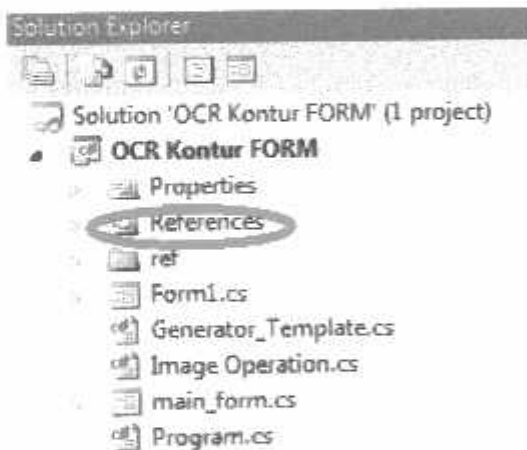
Berikut implementasi pada kode C#:

```
templates.Sort(new Comparison<FoundTemplateDesc>((t1, t2) => -  
t1.sample.contour.SourceBoundingRect.Area().CompareTo(t2.sample.contour.SourceBoundingRect.Area())));
```

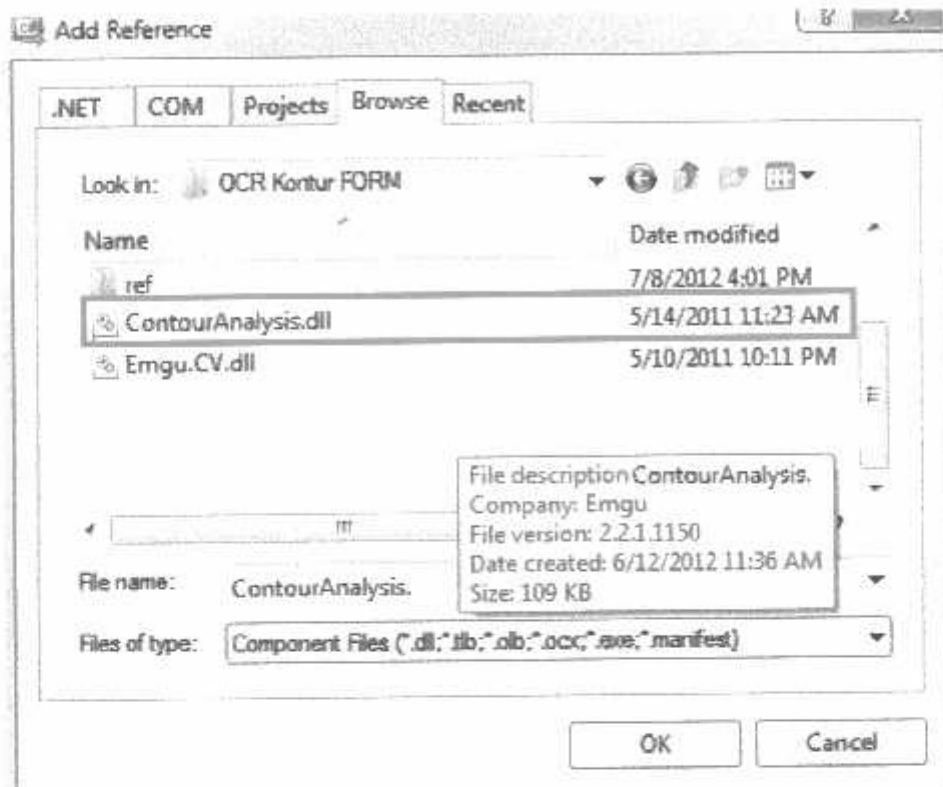
### 3.3.4 Menambahkan Library ConturAnalys pada Aplikasi OCR Kontur

Untuk dapat menggunakan berbagai fungsi pada library ContourAnalys kita harus menambah library tersebut ke dalam aplikasi yang kita buat. Pada subbab ini akan dijelaskan tahapan yang dilakukan untuk dapat menggunakan fungsi library pada sebuah program.

1. Pada jendela Solution Explorer Project yang kita buat klik kanan Reference.



2. Lalu klik Add References > Browse> buka direktori penyimpanan folder library > pilih ContourAnalysis.dll lalu klik OK



3. Walaupun kita telah menambah library pada Reference, tidak langsung dapat menggunakan fungsi library. Untuk itu perlu mendeklarasikan namespace dari library ContourAnalysis pada script/ code aplikasi yang kita buat yaitu seperti dibawah ini.

```

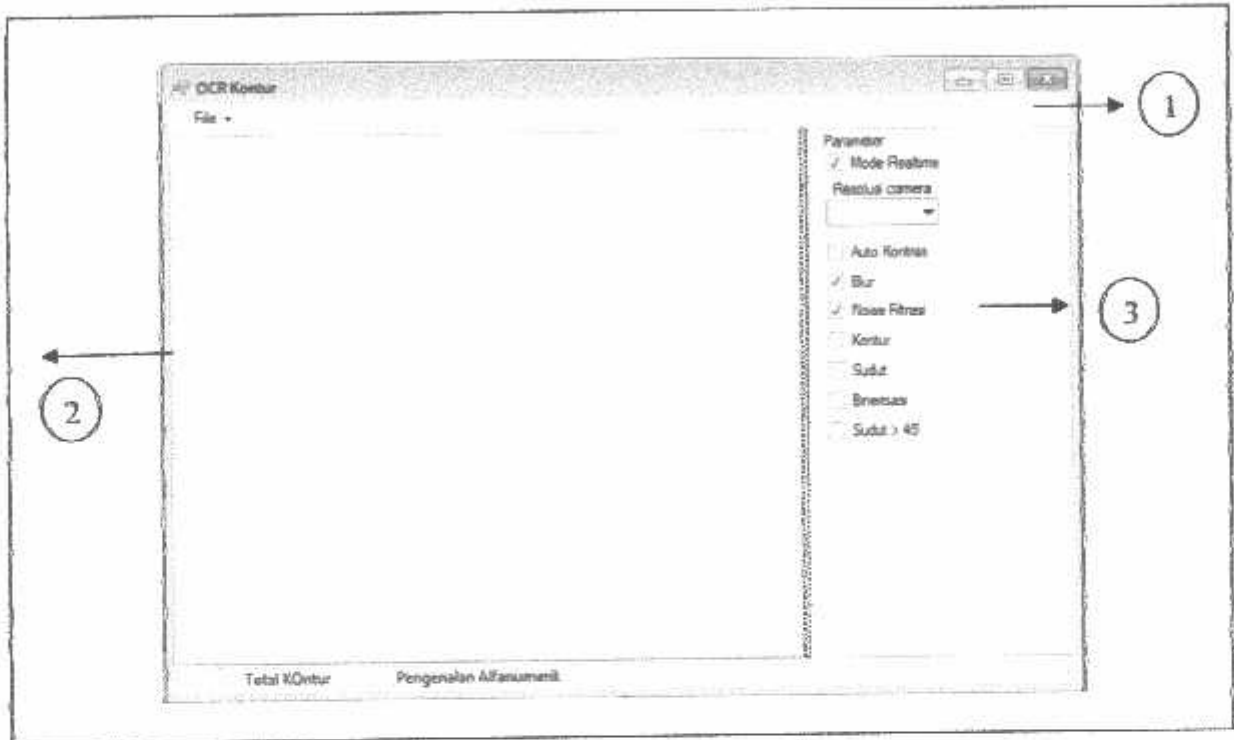
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Emgu.CV;
using Emgu.CV.Structure;
using ContourAnalysisNS;
using System.Drawing;
using System.Threading.Tasks;

namespace OCR_Kontur
{
    public class Image_Operation
    {
    }
}

```

### 3.3.5 Perancangan Antar Muka (*Interface*)

Perancangan *interface* adalah bagian yang penting dalam aplikasi, karena yang pertama kali dilihat ketika aplikasi dijalankan adalah tampilan antar muka (*interface*) aplikasi. Perancangan Antarmuka Main Form yaitu:



Gambar 7 Rancangan Interface

Keterangan Gambar:

1. Toolstrip

Akan dibuat dengan berisikan toolbar yang berfungsi sebagai menu untuk membuka gambar yang telah di scan dari file.

2. Display

Akan di buat tampilan baik file citra digital yang akan di kenali karakternya maupun hasil dari pengenalan gambar, kemudian akan dibuat juga tampilan *realtime* pengenalan di bagian ini. Display ini dibuat dengan memanfaatkan **ImageBox** pada library Emgu.CV.

### 3. Groupbox

Akan dibuat beberapa fitur didalamnya yang berfungsi sebagai beberapa tool utama dalam proses Pengenalan Gambar, beberapa fitur tersebut dibuat dengan memanfaatkan toolbox yaitu **Checkbox,ComboBox,Numeric**



## BAB IV

### *Implementasi Dan Pengujian Sistem*

Pada bab ini akan dijelaskan tahap pembuatan dari rancangan aplikasi **OCR Kontur**, sehingga menjadi sebuah aplikasi yang utuh dan dapat digunakan oleh pengguna.

#### **4.1 Spesifikasi Perangkat Keras Dan Lunak**

*Dalam menerapkan rancangan yang telah dibuat, ada beberapa hal yang harus dibutuhkan. Perangkat keras dan perangkat lunak merupakan 2 hal yang selalu dibutuhkan dalam mengimplementasikan rancangan yang telah ada.*

##### **4.1.1 Spesifikasi Perangkat Keras**

Dalam penerapan dari rancangan yang telah dijelaskan sebelumnya dibutuhkan beberapa perangkat keras untuk menyajikan aplikasi ini. Adapun alat-alat yang dibutuhkan adalah:

##### **1. PC berbasis Sistem Operasi Windows**

dengan spesifikasi sebagai berikut :

- Sistem Operasi : Windows 7 Ultimate 64-bit (6.1, build 7600)
- Processor : intel(R) core (TM) i7-2670QM CPU @2.20 Ghz
- Memory : 4096 MB RAM
- VGA : 251MB
- Webcam

#### 4.1.2 Spesifikasi Perangkat Lunak

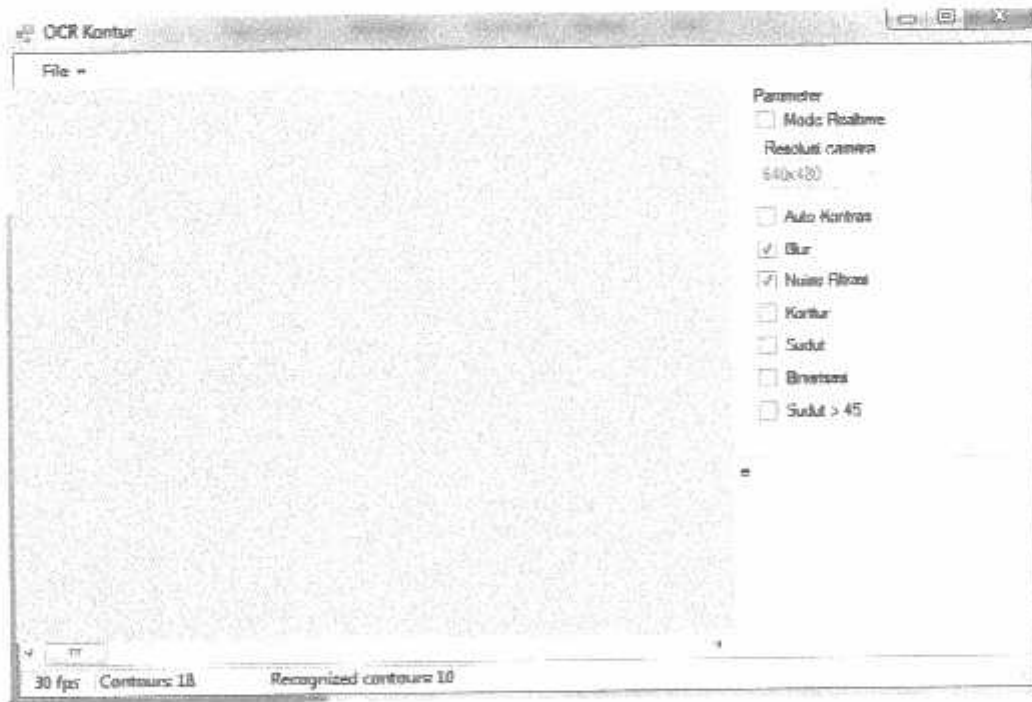
Dalam menerapkan rancangan yang telah dibuat, dibutuhkan beberapa *software* untuk membuat program aplikasi **OCR Kontur** yaitu :

1. Microsoft Visual C# 2010 Express Dalam hal ini digunakan Bahasa pemrograman C# dan library Emgu.CV untuk operasi kontur.
2. Sistem Operasi

Sistem Operasi yang digunakan adalah Windows 7 (64-bit)

#### 4.2 Implementasi Aplikasi OCR Kontur

Aplikasi yang dibangun bisa dijalankan di operasi windows apapun dengan syarat telah terinstal Net.Framework 4. Apabila di sebuah PC tidak terdapat webcam, program menggunakan mode realtime tidak dapat dijalankan.

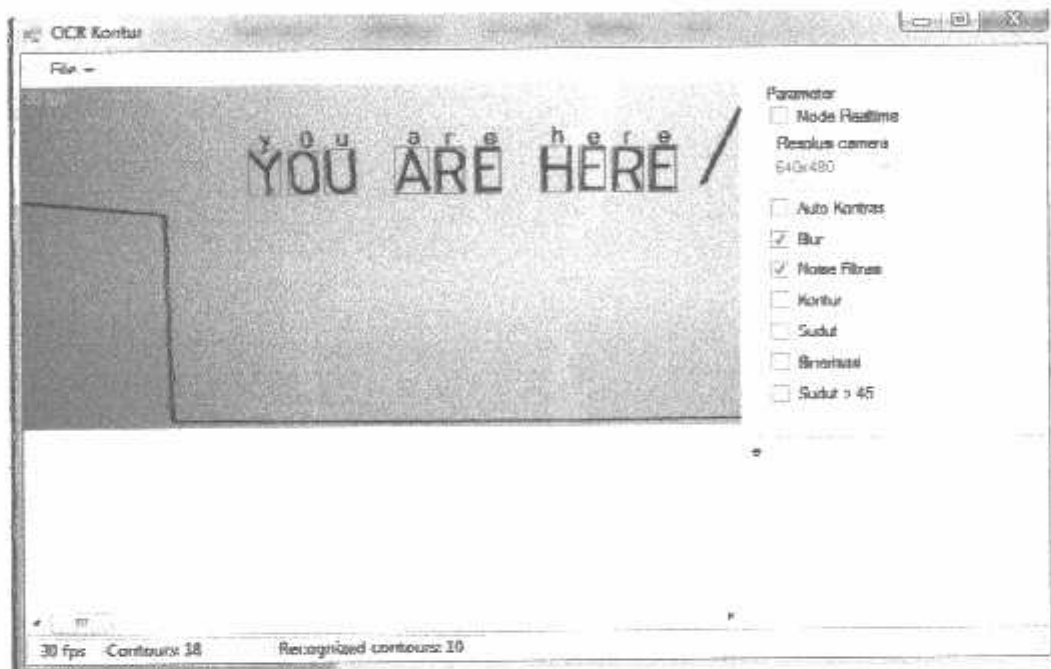


Gambar 8 Tampilan Aplikasi OCR Kontur

### 4.3 Pengujian Sistem

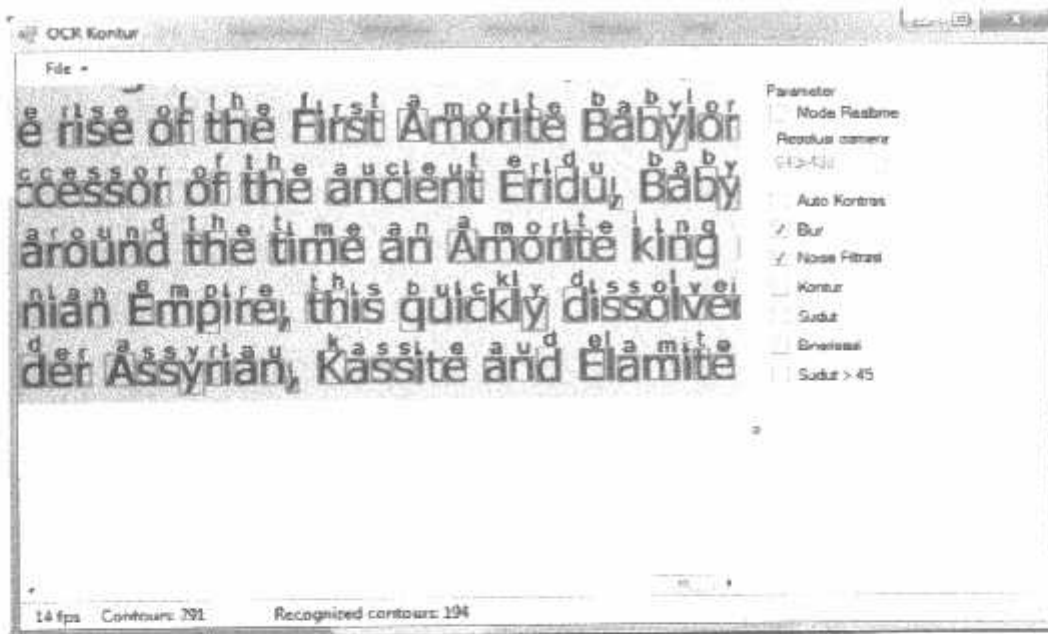
Pada bagian ini akan di dilakukan pengujian terhadap Aplikasi **OCR Kontur** yang telah dibuat pengujian dilakukan menggunakan file teks cetak berbentuk JPEG untuk mengetahui berapa pesan tingkat keberhasilan aplikasi dalam mengenali huruf image file tersebut.

Pengujian pertama menggunakan gambar yang sederhana terlebih dahulu. Berikut contohnya:



Gambar 9 Pungujian citra digital pada objek scdcrhana

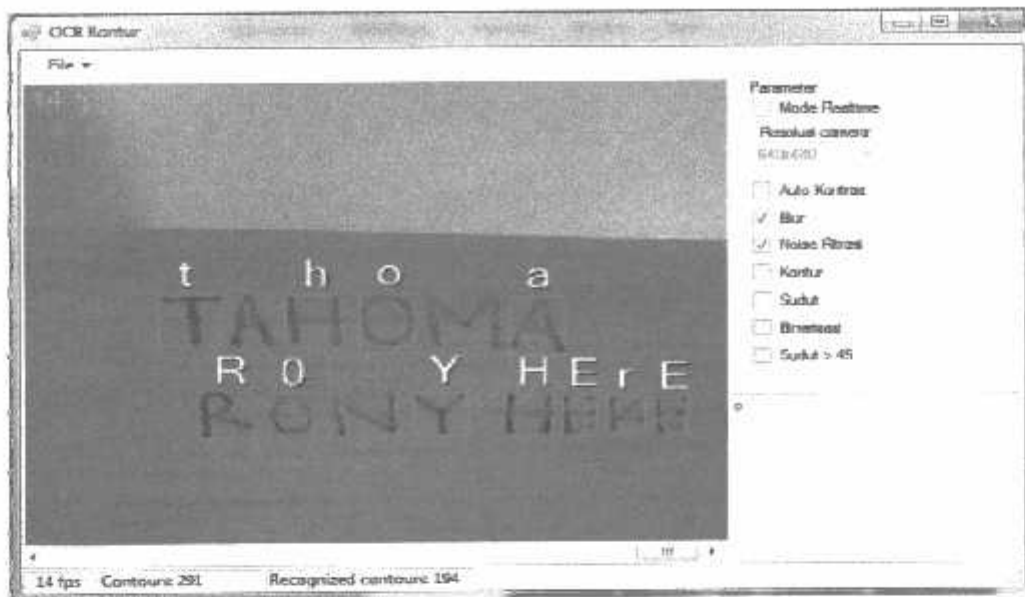
Dari hasil pengujian diatas dapat dilihat bahwa sistem yang di buat mampu mengenali seluruh karakter huruf yang ada.dengan kecepatan tinggi.Pengujian kedua menggunakan gambar yang lebih kompleks, berikut contohnya:



Gambar 10 Pengujian citra digital pada objek kompleks

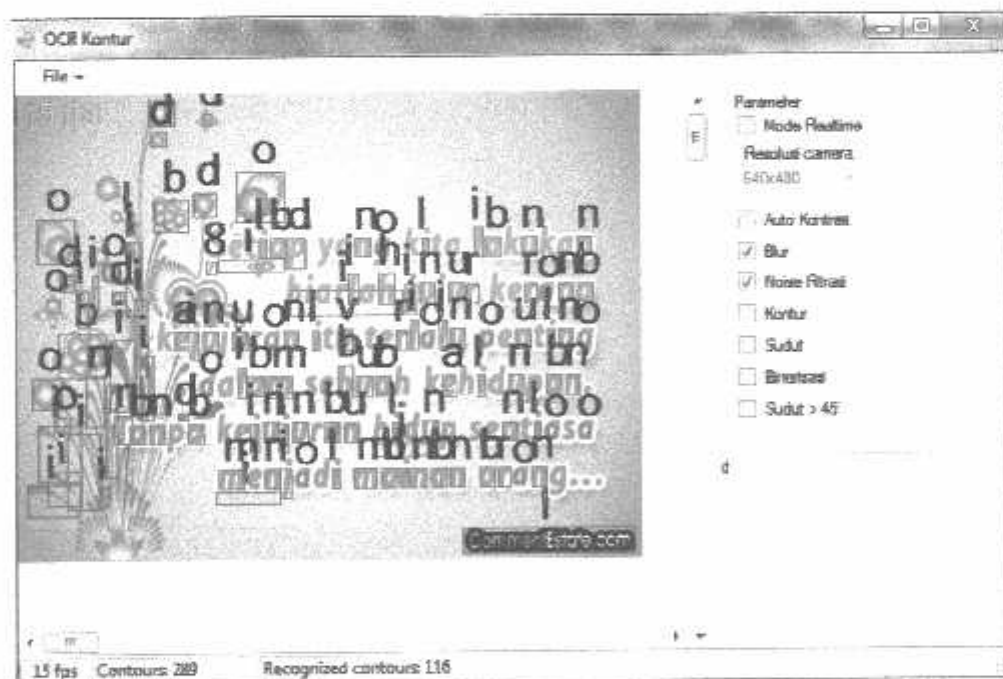
Dari hasil pengujian di atas dapat di lihat bahwa system yang di buat mampumengenali hamper seluruh karakter huruf sekitar 97% dengan kecepatan tertinggi.

Selain kedua pengujian di atas kita jg akan melakukan pengujian terhadap system realtime menggunakan webcam. Berikut contohnya:



Gambar 11 Hasil pengujian teks cetak realtime Mode

Tidak seperti menggunakan file image yang telah tersimpan dari hasil pengujian diatas terlihat bahwa persentase keberhasilan lebih kecil. Walaupun demikian eberhasilannya diatas 50%.



Gambar 12 Hasil pengujian teks cetak realtime Mode

Dari hasil pengujian di atas dapat di lihat bahwa system yang di buat tidak bisa mampumengenali seluruh karakter huruf,karna pada program ini tulisan yang bisa di kenali hanya tulisan TAHOMA saja.

## BAB V

### *Kesimpulan*

#### 5.1 Kesimpulan

*Dari hasil pengembangan aplikasi Implementasi Algoritma Contour Analys Untuk Pengenalan Objek Alfanumerik Menggunakan C# dapat diambil beberapa kesimpulan, yaitu :*

1. *Algoritma Contour Analys dengan menggunakan Library Contour Analys dapat melakukan pengenalan objek Alfanumerik dengan cepat dan dengan tingkat keakuratan yang cukup tinggi.*
2. *Karena aplikasi dibuat menggunakan Visual Studio, jadi aplikasi ini berbasis windows sehingga sangat *user friendly*.*
3. *Hasil pengenalan objek alfanumerik hanya berupa tampilan pada image box.*

#### 5.2 Saran

Saran yang dapat diberikan penulis atas penelitian ini adalah sebagai berikut:

1. *Aplikasi ini perlu dikembangkan lebih lanjut untuk mencapai kesempurnaan dalam pengenalan objek alfanumerik, terutama pada metode realtime menggunakan webcam. Adanya pergerakan objek membuat tidak stabil sehingga mempengaruhi ketepatan dalam pengenalannya.*
2. *Karena hasil hanya berupa tampilan pada program maka untuk selanjutnya diharapkan dapat menghasilkan tampilan berupa teks sehingga mencanpai kesempurnaan dalam proses pengalihan dokumen menjadi data digital.*

## Daftar Pustaka

1. Anil, K., Jain (1989). *Fundamentals of Digital Image Processing*. Prentice-Hall, Inc. New Jersey.
2. Fairhurst, Michael C. (1988) "*Computer vision for robotic systems : An introduction*" Prentice Hall.
3. Fuhrman JA (2000), "Introduction to the contour analysis".
4. Raharjo, Andi W., Arie Hartanto, (2008) "Pengembangan Aplikasi Pengenalan Karakter Alfanumerik Dengan Menggunakan Algoritma *Neural Network Three-Layer Backpropagation*".
5. Schalkoff, Robert J. (1989) "*Digital image processing and computer vision*". John Wiley & Sons.
6. Nash, Trey. (2011) "Accelerated c# 2010". Appres.
7. [http://en.wikipedia.org/wiki/Optical\\_character\\_recognition](http://en.wikipedia.org/wiki/Optical_character_recognition)
8. [www.csharp-indonesia.com](http://www.csharp-indonesia.com)
9. <http://www.emgu.com>



## FORMULIR PERBAIKAN UJIAN SKRIPSI

Dalam pelaksanaan ujian skripsi jenjang Strata I Program Studi Teknik Elektro Konsentrasi Teknik Komputer dan Informatika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : RONY HAMKA  
NIM : 05.12.669  
PROGRAM STUDI : Teknik Komputer dan Informatika S-1  
JUDUL : IMPLEMENTASI ALGORITMA COUNTOUR ANALYS  
UNTUK PENGENALAN OBJEK ALFANUMERIK  
MENGGUNAKAN C#

No	Penguji	Tanggal	Uraian	Paraf
1.	Penguji I	08 agustus 2012	1. Penulisan di rapikan 2. Diagarm sistem perlu di perjelas 3. Daptar pustaka di lengkapi yang sesuai	
2.	Penguji II	08 agustus 2012	1. Pengujian di perbayak lalu di buatkan tabel	

Disetujui :

Dosen Penguji I

Dr. Eng. Aryuanto Soetedjo, ST, MT  
NIP.P. 1030800417

Dosen Penguji II

M. Ibrahim Ashari, ST, MT  
NIP.P. 1030100358

Mengetahui :

Dosen Pembimbing I

Irmalia Suryani Faradisa, ST, MT  
NIP.P. 1030000365

Dosen Pembimbing II

Sandy Nataly Manja, S.KOM  
NIP.P. 1030800418





## FORMULIR PERBAIKAN UJIAN SKRIPSI

Dalam pelaksanaan ujian skripsi jenjang Strata 1 Program Studi Teknik Elektro Konsentrasi Teknik Komputer dan Informatika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : RONY HAMKA  
NIM : 05.12.669  
PROGRAM STUDI : Teknik Komputer dan Informatika S-1  
JUDUL : IMPLEMENTASI ALGORITMA COUNTOUR ANALYS  
UNTUK PENGENALAN OBJEK ALFANUMERIK  
MENGUNAKAN C#

No	Penguji	Tanggal	Uraian	Paraf
1.	Penguji I	08 agustus 2012	1. Penulisan di rapikan 2. Diagarm sistem perlu di perjelas 3. Daptar pustaka di lengkapi yang sesuai	
2.	Penguji II	08 agustus 2012	1. Pengujian di perbayak lalu di buatkan tabel	

Disetujui :

Dosen Penguji I

Dr.Eng.Aryuanto Soetedjo,ST,MT  
NIP.P. 1030800417

Dosen Penguji II

M. Ibrahim Ashari, ST, MT  
NIP.P. 1030100358

Mengetahui :

Dosen Pembimbing I

Irmalia Suryani Faradisa, ST, MT  
NIP.P. 1030000365

Dosen Pembimbing II

Sandy Nataly Manja,S.KOM  
NIP.P. 1030800418

## Lampiran

### From Template

```
using System.Drawing;
using Emgu.CV.Structure;
using Emgu.CV;

namespace OCR_Kontur_FORM
{
    class Generator_Template
    {
        public static void GenerateChars(Image_Operation processor, char[]
chars, Font font)
        {
            Bitmap bmp = new Bitmap(400, 400);
            font = new Font(font.FontFamily, 140, font.Style);
            Graphics gr = Graphics.FromImage(bmp);
            //
            processor.onlyFindContours = true;
            foreach (char c in chars)
            {
                gr.Clear(Color.White);
                gr.DrawString(c.ToString(), font, Brushes.Black, 5, 5);
                GenerateTemplate(processor, bmp, c.ToString());
            }
            processor.onlyFindContours = false;
        }

        public static void GenerateAntipatterns(Image_Operation processor)
        {
            Bitmap bmp = new Bitmap(200, 200);
            Graphics gr = Graphics.FromImage(bmp);
            //
            processor.onlyFindContours = true;
            //square
            gr.Clear(Color.White);
            gr.FillRectangle(Brushes.Black, new Rectangle(10, 10, 80, 80));
            GenerateTemplate(processor, bmp, "antipattern");
            //rect1
            gr.Clear(Color.White);
            gr.FillRectangle(Brushes.Black, new Rectangle(10, 10, 50,
100));
            GenerateTemplate(processor, bmp, "antipattern");
            //rect2
            gr.Clear(Color.White);
            gr.FillRectangle(Brushes.Black, new Rectangle(10, 10, 20,
100));
            GenerateTemplate(processor, bmp, "antipattern");
            //circle
            gr.Clear(Color.White);
            gr.FillEllipse(Brushes.Black, new Rectangle(10, 10, 100, 100));
            GenerateTemplate(processor, bmp, "antipattern");

            processor.onlyFindContours = false;
        }

        private static void GenerateTemplate(Image_Operation processor,
Bitmap bmp, string name)
        {
            processor.ProcessImage(new Image<Bgr, byte>(bmp));
        }
    }
}
```

```

        //find max contour
        if (processor.samples.Count > 0)
        {
            processor.samples.Sort((t1, t2) => -
                t1.sourceArea.CompareTo(t2.sourceArea));
            processor.samples[0].name = name;
            processor.templates.Add(processor.samples[0]);
        }
    }
}

```

## Image Opration

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Emgu.CV;
using Emgu.CV.Structure;
using System.Drawing;
using System.Threading.Tasks;
using ContourAnalysisMS;

namespace OCR_Kontur_FORM
{
    class Image_Operation
    {
        //settings
        public bool equalizeHist = false;
        public bool noiseFilter = false;
        public int cannyThreshold = 50;
        public bool blur = true;
        public int adaptiveThresholdBlockSize = 4;
        public double adaptiveThresholdParameter = 1.2d;
        public bool addCanny = true;
        public bool filterContoursBySize = true;
        public bool onlyFindContours = false;
        public int minContourLength = 15;
        public int minContourArea = 10;
        public double minFormFactor = 0.5;
        //
        public List<Contour<Point>> contours;
        public Templates templates = new Templates();
        public Templates samples = new Templates();
        public List<FoundTemplateDesc> foundTemplates = new
        List<FoundTemplateDesc>();
        public TemplateFinder finder = new TemplateFinder();
        public Image<Gray, byte> binarizedFrame;

        public void ProcessImage(Image<Bgr, byte> frame)
        {
            ProcessImage(frame.Convert<Gray, Byte>());
        }

        public void ProcessImage(Image<Gray, byte> grayFrame)
        {

```

```

    if (equalizeHist)
        grayFrame._EqualizeHist();//autocontrast
    //proses smooth atau penghalusan citra
    Image<Gray, byte> smoothedGrayFrame = grayFrame.PyrDown();
    smoothedGrayFrame = smoothedGrayFrame.PyrUp();
    //Poses treshold
    Image<Gray, byte> cannyFrame = null;
    if (noiseFilter)
        cannyFrame = smoothedGrayFrame.Canny(new
Gray(cannyThreshold), new Gray(cannyThreshold));
    //blur
    if (blur)
        grayFrame = smoothedGrayFrame;
    //binerisasi
    CvInvoke.cvAdaptiveThreshold(grayFrame, grayFrame, 255,
Emgu.CV.CvEnum.ADAPTIVE_THRESHOLD_TYPE.CV_ADAPTIVE_THRESH_MEAN_C,
Emgu.CV.CvEnum.THRESH.CV_THRESH_BINARY, adaptiveThresholdBlockSize +
adaptiveThresholdBlockSize % 2 + 1, adaptiveThresholdParameter);
    //
    grayFrame._Not();
    //
    if (addCanny)
        if (cannyFrame != null)
            grayFrame._Or(cannyFrame);
    //
    this.binarizedFrame = grayFrame;

    //dilate canny contours for filtering
    if (cannyFrame != null)
        cannyFrame = cannyFrame.Dilate(3);

    //find contours
    var sourceContours =
grayFrame.FindContours(Emgu.CV.CvEnum.CHAIN_APPROX_METHOD.CV_CHAIN_APPROX_N
ONE, Emgu.CV.CvEnum.RETR_TYPE.CV_RETR_LIST);
    //filter contours
    contours = FilterContours(sourceContours, cannyFrame,
grayFrame.Width, grayFrame.Height);
    //find templates
    lock (foundTemplates)
        foundTemplates.Clear();
    samples.Clear();

    lock (templates)
        Parallel.ForEach<Contour<Point>>(contours, (contour) =>
        {
            var arr = contour.ToArray();
            Template sample = new Template(arr, contour.Area,
samples.templateSize);
            lock (samples)
                samples.Add(sample);

            if (!onlyFindContours)
            {
                FoundTemplateDesc desc =
finder.FindTemplate(templates, sample);

                if (desc != null)
                    lock (foundTemplates)
                        foundTemplates.Add(desc);
            }
        });

```

```

        }
    }
}
//
FilterByIntersection(ref foundTemplates);
}

private static void FilterByIntersection(ref
List<FoundTemplateDesc> templates)
{
    //sort by area
    templates.Sort(new Comparison<FoundTemplateDesc>((t1, t2) => -
t1.sample.contour.SourceBoundingRect.Area().CompareTo(t2.sample.contour.Sou
rceBoundingRect.Area())));
    //exclude templates inside other templates
    HashSet<int> toDel = new HashSet<int>();
    for (int i = 0; i < templates.Count; i++)
    {
        if (toDel.Contains(i))
            continue;
        Rectangle bigRect =
templates[i].sample.contour.SourceBoundingRect;
        int bigArea =
templates[i].sample.contour.SourceBoundingRect.Area();
        bigRect.Inflate(4, 4);
        for (int j = i + 1; j < templates.Count; j++)
        {
            if
(bigRect.Contains(templates[j].sample.contour.SourceBoundingRect))
            {
                double a =
templates[j].sample.contour.SourceBoundingRect.Area();
                if (a / bigArea > 0.9d)
                {
                    //pilih template berdasarkan rate
                    if (templates[i].rate > templates[j].rate)
                        toDel.Add(j);
                    else
                        toDel.Add(i);
                }
                else//hapus tempate
                    toDel.Add(j);
            }
        }
    }
    List<FoundTemplateDesc> newTemplates = new
List<FoundTemplateDesc>();
    for (int i = 0; i < templates.Count; i++)
        if (!toDel.Contains(i))
            newTemplates.Add(templates[i]);
    templates = newTemplates;
}

private List<Contour<Point>> FilterContours(Contour<Point>
contours, Image<Gray, byte> cannyFrame, int frameWidth, int frameHeight)
{
    int maxArea = frameWidth * frameHeight / 5;
    var c = contours;
    List<Contour<Point>> result = new List<Contour<Point>>();
    while (c != null)

```

```

    {
        if (filterContoursBySize)
            if (c.Total < minContourLength ||
                c.Area < minContourArea || c.Area > maxArea ||
                c.Area / c.Total <= minFormFactor)
                goto next;

            if (noiseFilter)
            {
                Point p1 = c[0];
                Point p2 = c[(c.Total / 2) % c.Total];
                if (cannyFrame[p1].Intensity <= double.Epsilon &&
                    cannyFrame[p2].Intensity <= double.Epsilon)
                    goto next;
            }
            result.Add(c);

        next:
            c = c.HNext;
    }
    return result;
}
}
}

```

### Program

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;

namespace OCR_Kontur_FORM
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new MainForm());
        }
    }
}

```



INSTITUT TEKNOLOGI NASIONAL MALANG  
FAKULTAS TEKNOLOGI INDUSTRI  
JURUSAN TEKNIK ELEKTRO

### Formulir Perbaikan Ujian Skripsi

Dalam pelaksanaan Ujian Skripsi Janjang Strata 1 Jurusan Teknik Elektro Konsentrasi T. Energi Listrik / T. Elektronika / T. Infokom, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : Ronyblanka  
NIM : 0512669  
Perbaikan melalui :

- <sup>Diagram</sup> Perhitungan beban dirapikan (gambar 4, 5)
- Diagram blok sistem perlu di perjelas (gambar 3)
- Daftar pustaka di lengkapi yg sesuai.

Malang, 8/8/12

Aryuanto

### Formulir Perbaikan Ujian Skripsi

Dalam pelaksanaan Ujian Skripsi Janjang Strata 1 Jurusan Teknik Elektro Konsentrasi T. Energi Listrik / T. Elektronika / T. Infokom, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : Rony Hamka  
NIM : 0512669  
Perbaikan meliputi :

Terdapat pengisian di paragraf dengan skema  
50% dan 77%  
Pengisiannya diperbaiki, kemudian  
ditambahkan tabel.

Malang, 2 agust 2012

  
M. Ibrahim



# SKRIPSI

## IMPLEMENTASI ALGORITMA COUNTOUR ANALYS UNTUK PENGENALAN OBJEK ALFANUMERIK MENGGUNAKAN C#



*Disusun Oleh :*

**RONY HAMKA**  
**NIM: 05.12.669**

**KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA**  
**JURUSAN TEKNIK ELEKTRO S-1**  
**FAKULTAS TEKNOLOGI INDUSTRI**  
**INSTITUT TEKNOLOGI NASIONAL MALANG**  
**2012**

---